NewIdPAttribute

Accessing and Releasing an Attribute from the IdP

Follow these steps to pull in and release an attribute from the IdP.

1. Pull in Raw Attributes

Shibboleth doesn't store any attributes about users itself; it relies on external data stores to supply user information to be released. These attributes are pulled from data sources using data connectors.

- The static data connector is used to add an attribute and a common value to every person served by the identity provider. An example usage of this connector would be to add an entitlement attribute that everyone in your organization recieves.
- The relational database connector is used to pull attributes from a relational database by executing some configured SQL.
- The LDAP connector is used to pull attributes from an LDAP directory by executing an LDAP filter on a specific branch.

If none of these connectors meets your need, you may create a custom data connector.

2. Prepare Attributes

After defining a data connector, the IdP next needs to query the data source for attribute information. Once this attribute has been pulled in, it's transformed into a SAML attribute for transport to the SP. Along the way, they can be transformed, filtered, composed, split, and more. The rules that govern this process are the AttributeDefinitions.

- The simple attribute definition allows raw attributes to be used as is or to make an attribute available under a different name.
- The composite attribute definition allows multiple raw attributes to be composed with static data or other attributes. This could be used to create a "FullName" attribute by concatenating a first and last name attribute or to add a URN namespace to every group identifier a user is in, for example.
- The regular expression attribute definition allows text matching a particular regular expression to be replaced with other text.
- The scriptlet attribute definition can execute Java code, given in its configuration, upon an attribute. Anything that can be expressed through Java code could be done here.
- The SAML2 persistent ID attribute definition produces SAML 2 compliant persistent NameIDs used as attribute values.

If none of these definitions meet your need you may create a custom attribute definition.

3. Release Attributes

Once all of your attributes are defined you need to define a policy to release them to one or more service providers. Here's how to write a rule to release an attribute.

4. Testing Your Configuration

There are two tests available to ensure your attributes are being properly generated and released.

- The attribute resolution test allows you to determine if your attribute definitions are working properly.
- The attribute release test allows you to determing if your attribute release policy will provide the expected attributes to a given service provider.